Models

2

In the previous chapter we outlined the various aspects of Business Process Management. Since models are often used in practice to describe these aspects, the following sections will first look in more detail at the tasks and properties of models and modeling. We then present examples of models from various areas of expertise that in our experience have proven helpful in numerous Business Process Management projects. There is no claim to completeness for this exposition, but it can serve as an orientation for the reader when needing to choose description models for individual project needs.

2.1 Model and Reality

"You don't have to understand the world, you just have to find your way around in it." According to internet sources, this sentence is attributed to Albert Einstein. Who understands what is going on in the world? Who knows how it works? Therefore, we should take care of our world, namely the part of the world that is important to us at the moment. We should recognize that we create or construct our world on a daily basis. Any excerpt of reality is naturally determined by our subjective interests. We decide which part of the world we want to consider and which aspects seem important to us.

In doing so, we identify the artifacts and the relationships between them that are essential for us. Such an abstraction of a part of reality is called a model. It is also possible that the segment of reality considered is already a model itself. This allows parts of an already existing model to be examined more closely. This would then be a model of a model. This process can be repeated as many times as desired.

Every person has his or her own subjective view of the world or a part of it. Different people may consider the same part of reality and arrive at different models because they set different priorities (cf., Fig. 2.1).

Since models do not cover all aspects of the assigned reality, there are indeed incidents in reality that are not covered by a model and are incomprehensible therein.

© The Author(s) 2020

A. Fleischmann et al., *Contextual Process Digitalization*,
https://doi.org/10.1007/978-3-030-38300-8_2

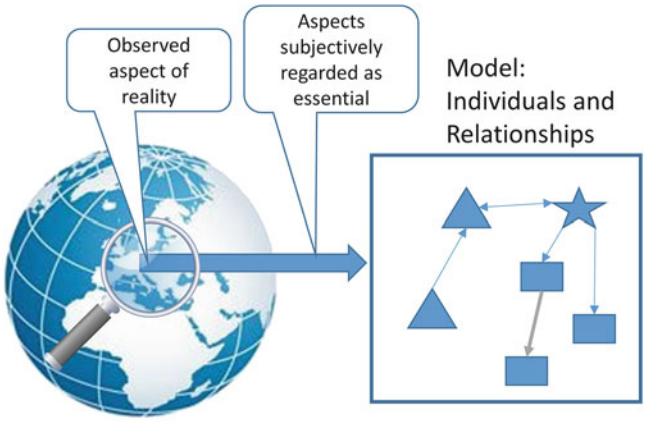

Figure 2.1: Modeling

These phenomena, which cannot be explained in a model itself, form its limits. The creator of a model may decide to adapt it accordingly when intending to include phenomena not yet covered or to omit these if they should no longer be taken into consideration. In any case, a model remains a simplification of reality, which is why there will always be phenomena that are not covered by a model. Each person needs to decide whether the model in use is sufficient for his purposes. If one wants to cover all aspects of reality with a single model, it reaches the complexity of reality. Then one doesn't understand the model to the same extent one didn't understand the reality before mapping it to a model. The intention of modeling, namely, to make reality more comprehensible and manageable, is then no longer achieved.

The creation of a model and its use are subjective activities, i.e., the creator chooses the characteristics of the representation of reality according to his ideas. However, it is common for groups of acting participants, subsequently referred to as subjects or actors, to agree on a model for observing reality. Many scientific schools of thought are based on such common models of the involved researchers.

Modeling is an essential activity in all sciences, be it philosophy, sociology, physics, chemistry, all engineering sciences, economics, etc. The respective models have different tasks: Either they depict the specific part of reality under consideration, as is traditionally done in the natural sciences, or they serve to try out certain necessary changes to said part (simulation model). This is particularly important to avoid endangering human life. Before going into series production, the properties of cars are checked in safety models in appropriate tests. This does not happen with people, but instead with models of people, so-called dummies. In doing so, two types of models are combined with each other: The car model, which implements the corresponding safety concepts, and the human model (dummy) to investigate the risks of injury.

The state of affairs captured in a model can be adapted as often as required until a desired result is achieved for the phenomena considered in the model. For instance,

further safety concepts are added to the safety model of a car until the desired reduction of the risk of injury is achieved. Such models are then no longer images of reality, but rather represent a desired reality. The desired properties are then transferred back into reality, e.g., by incorporating the safety concepts tested in the model into the series production of cars.

The aim of modeling is always to find our way around in the world, or to try out safely how a corresponding change in reality would affect us. We will not likely succeed in understanding what holds the world together at its core in the foreseeable future. The corresponding model would then be the world that does not exist [1].

2.2 Properties of Models

Models serve both as the abstract representation of the observed reality in the sense of a cognitive function and as the design of the observed reality in the sense of a conclusion. As already mentioned, once a model is modified until it corresponds to a desired reality, it can be used as a blueprint for a corresponding transformation of reality. In natural sciences such as physics, models predominantly have a cognitive function, while in engineering sciences and business administration they are intended to support the shaping of reality [2].

In clarifying the model concept used thus far we follow the studies of Herbert Stachowiak. In his model theory he examined the characteristics of models and the properties derived from these more closely [3]. Accordingly, models are identified by at least three characteristics ([3] page 131):

1. Mapping

   A model is always a representation of something. It can be a depiction or representation of a natural or artificial original, whereby this original itself can again be a model. The originals can be created in a natural way, technically produced, or simply exist in some other way. Models can be described or represented in very different ways:
   - Mental models: Imagination in the human mind
   - Verbal models: Natural language description
   - Graphical models: Technical drawings or other pictures
   - Material models: Models of buildings
   - Formal models: Mathematical models, computer programs, etc.

A model and the original form a class of attributes. Attributes are characteristics and properties of individuals, relations between individuals, properties of properties, properties of relations, etc. Stachowiak leaves it up to the modeling subject to determine how to conceptualize individuals. He considers individuals as attributes on level 0. Sets of attributes can be combined to form classes, which then form attributes on level 1. These classes can be combined again to form a next attribute level, and so on.

2. Reduction

   In general, a model does not capture all attributes of the original, but only those that appear relevant to the creator or user of the model.

   Since not all attributes of the original are captured by a model, a pragmatic dimension has been introduced in the broader sense. In the "broader sense" here means that not yet specific pragmatic-operational aspects are considered, according to which the attribute classes that are to be included in a model are selected. This initial selection of attributes is intuitive and arbitrary. In the narrower sense, the reduction is pragmatic only when the intentions and operational objectives of the model creator or user influence the selection of model-relevant attributes. These adjustments to the intended practical use are made in the next step.

3. Pragmatism

   After the intuitive selection of the attributes, it is checked whether the intended purpose has been achieved. Models are not clearly assigned to their originals. They serve as a replacement function

   - for certain discerning or acting model-using subjects (for whom?).

     Models are not only models of something, they are also models for someone. This someone can be a human being or an artificial model user such as a computer program. For a modeler, models can serve as a possibility to find one's way around in the world, i.e., the modeler is also a user of the model. Modelers and model users can also be two different subjects.
   - within certain time intervals (when?)

     Models also perform a function over time, i.e., their use is related to a specific point in time or to a defined time interval. During this time, the observed reality or the ideas of the modeler or model user may have changed in such a way that additional attributes should become part of the model.
   - with restriction to certain mental or actual operations (why?).

     Models are created for a certain purpose, be it for better understanding a certain part of reality or for creating a blueprint for the transformation of reality.

When creating a model, modelers are always in a certain dilemma. On the one hand, the model should sufficiently reflect the desired aspects of reality, whereby it is not clearly defined what is sufficient. On the other hand, the model should not be too complex in order to remain manageable. This conflict of objectives leads to the fact that most models are developed iteratively until they reach the end of their life cycle due to increasing complexity - they are no longer manageable.

In the following sections, we present examples of models that consider aspects that are explicitly or implicitly incorporated into business process models. We have grouped the examples into models from the social sciences, business administration, business informatics, and computer science. The classification into these groups is not entirely free of overlaps, since especially business informatics as a cross-sectoral discipline considers issues from various perspectives.

2.3 Models of the Social Sciences

Business Process Management has to do with people and machines. It aims at organizing their interaction while taking into account additional requirements with regards to technical, economic and ecological feasibility. In particular, the interaction and coexistence of people has been a topic of philosophy for thousands of years. Philosophy, as the doctrine of the basic rules and structures of life, the world and knowledge, seeks to fathom, interpret and understand the world and human existence. The original meaning of philosophy was the teaching of good life.

In this sense philosophy is the attempt to create the comprehensive model of our world, but this has not yet been successful. Social philosophers reduce the view and try to create a model of society as some part of reality and thus better understand its meaning and essence. In particular, social philosophies illuminate the relationship between the individual and the community as well as the structures of living together. They are therefore also regarded as variants of philosophy that touch on sociology. They should help sociologists to analyze social processes and support organizational developers in their work and help people to find their way around in the world. There are numerous organizational theories that focus on different aspects in their models (see [4–7]). The question of which organizational theory fits best cannot be answered. Organizational theories are models and thus, according to Stachowiak, the justified but subjective view of the modeler. Organizational theories emphasize the analysis of organizations in highly different ways and pursue different objectives. There are empirical studies on organizational theories that provide results in favor of, or against, a theory. However, the analysis methods used are controversial (see [8] page 68).

Organizational theories are based on certain perspectives on people, and the design of Business Process Management is strongly influenced by the prevailing image of people in an organization. Hence, in the following sections we discuss Taylorism, Habermas's Theory of Communicative Action and Luhmann's Social Systems as distinct organizational theories to exemplify different perspectives on human and organizational behavior.

Taylorism and Fordism

Taylorism introduced the ‘experiment’ into management theory and practice and is one of the classics of organizational theory. With the so-called Scientific Management, organizations received an instrument to design themselves efficiently. One important characteristic is the separation between planning mental work and performing manual work. According to Taylor’s view of man, workers are dumb and lazy and must therefore be subject to strict rules. Even today, this view often implicitly influences leadership behavior.

For Frederik Winslow Taylor, the most important goals of running a business were the perfection of the means of production and work processes, tighter organization and temporal structuring of workflows in the company, as well as a reorganization of the remuneration system. A core element of Taylorism is the design of work processes on the basis of time and movement studies. A breakdown of the

production process into the smallest work steps, an exoneration of workers from mental activities, as well as a change of the wage system should lead to an optimal use of existing performance potentials. The ultimate goal is to increase the productivity of human labor. This is done by dividing the work into its smallest units, which require only little or no cognitive effort to accomplish, and which can be repeated quickly and repetitively due to their small size or the content of the work.

Taylorism is based on the following core principles:

- Work scheduling
  The planning of the work is done by other persons than those who carry it out (separation of manual and mental work). In this way Taylor wanted to avoid the shirking that he accused the workers of. Through time and movement studies carried out by the mental workers, the least amount of movement and time required for a work step was to be determined.
- Incentivized wages
  These time and movement studies also revealed what the workers had to achieve in a certain period of time. A bonus ensured that the workers who were classified as dumb and lazy actually tried to achieve the specified performance.
- Selection of the most suitable workers
  One aim was to build up a first-class workforce by means of an appropriate selection mechanism. Tests were developed and used to identify particularly agile and nimble-fingered workers.
- Reconciliation between workers and management
  Taylor believed that the system he had developed could increase productivity to such an extent that the dispute over the distribution of profits would become a minor issue. This achievement should resolve the conflict between employers and employees.

Taylorism essentially considers the structuring of work steps but does not focus on their sequence. This aspect was addressed by Henry Ford, who coordinated the individual activities by introducing the assembly line. This step created the basis for the mass production that characterized the 20th century. The assembly line principle was also transferred to administration and strongly influenced Business Process Management. Flowcharts are the assembly line specifications for the execution of administrative tasks or the "production" of services.

Communicative Action According to Habermas

In contrast to Taylorism, Habermas's Theory of Communicative Action is based on insightful people who, through communication among themselves, come to a common rational action [9, 10]. Using his social model Habermas explains the processes in a society, such as the search for truth, for justice, etc. Consequently, it is a model that concerns everyone, because the issues of truth and justice affect all members of societies. The central aspect in Habermas's model of society is the so-called Communicative Action.

"Finally, the concept of Communicative Action refers to the interaction of at least two subjects capable of speaking and acting that enter into an interpersonal relationship (whether by verbal or nonverbal means). The actors seek an understanding of the action situation in order to coordinate their action plans, and thus their actions, amicably". (see [11], Volume 1, page 128).

According to Habermas, communication enables an individual human being, who is not gifted with rationality on his own initiative, to overcome this deficiency. Communication between people becomes intersubjective action and a possible source of rationality. Communicative Action means acting on the basis of mutual understanding between people.

Habermas wants to offer sociologists and politicians a model that they can use to analyze and shape society. The individual can use it to find his way around in today's societies, despite their complexity.

Social Systems According to Luhmann

Similar to Habermas's, Luhmann's Social Model is based on communication. The differences lie in the extent to which communication and action are combined. Luhmann only allows communication as a constituent aspect for organizations. Communication does not occur between people, but between at least two information-processing processors. Luhmann thus sees communication more abstractly. According to him, society does not consist of people or parts of people. Otherwise one would cut off something from society, when one cuts off something from a person. The body of a human being (i.e., as a biological system) with a conscious mind (a psychological system) is in many cases a prerequisite for the functioning of a social system, i.e., communication. However, a human being is not the social system itself. Luhmann makes no assertions about the nature of man in his organizational theory, thus leaving the perspective on humans open. People are only part of the organization insofar as they communicate with each other.

The communication between the information-processing processors consists of the so-called selections of information, message and understanding (see Fig. 2.2). The first two selections are for the sender and the third for the recipient. Communication as a piece with at least two actors in three acts is an indivisible unit, namely the smallest unit of a social system and the elementary operation of society

	Two information-processing processors, usually people or social systems	
	transmitter Luhmann's "Alter"	receiver Luhmann's "Ego"
Three selections:	1. Selection of information 2. Selection of message	3. Selection of acceptance/ understanding

Figure 2.2: Luhmann's understanding of communication

[12, 13]. This view can serve as a pattern for the definition of communication in business processes, completely independent of a specific perspective on humans.

Both Luhmann and Habermas put communication at the center of their organizational theory. The fact that these two important organizational theorists so strongly emphasize the communication aspect of the organization, and that their theories are widely accepted, can be seen as an indication for considering business processes as primarily communication oriented.

Organizations

Complex social systems can be divided into smaller social systems. This structure of a complex social system is called the organizational structure. The criteria according to which the division into smaller social systems takes place are subjective and depend on the respective intentions. According to Luhmann's definition of a social system, the individual social systems communicate within a more complex social system. Organizational structures are thus a model of a more complex social system.

In contrast to Luhmann's or Habermas's broad understanding of the term organization, a narrower understanding of organizations has also developed. In business administration, organization is the formal set of rules of a system based on the division of labor. In organizational sociology it refers to a special form of social entity that can be distinguished from other social entities such as families, groups, movements or networks. Essential characteristics of organizations are that people can join them or leave them. In addition, they have a purpose that they are geared to. Organizations have regulations on the division of labor, such as specialization according to performance, function, objects or space, or corresponding hybrid forms. This division of labor requires the coordination of individual activities. Hierarchy is the central instrument of coordination in organizational theory. The hierarchical coordination is supplemented by cadres, commissions, task forces, etc. For one-time problems or problems to be solved for the first time, the hierarchy is supplemented by a project organization.

2.4 Models of Business Administration

Business administration is the study of economic, organizational, technical and financial processes and structures in companies. Business Process Management is therefore also a part of business administration. Business processes serve to improve the economic efficiency of a company with all of the associated aspects such as customer satisfaction, employee motivation, the integration of partners, etc. For the structuring of all these aspects and for the analysis of their interaction, business administration has developed models which, when applied, have an effect on Business Process Management.

Business Model

A business model refers to the overall concept of a company. It represents the interrelationships as a model, how a company can generate added value for its

Figure 2.3: Schema of the Business Model Canvas

customers and thus achieve sustainable returns. In addition to the products and services offered, the focus is on the structure of the company, the definition of the target groups (customers) and how they are addressed, as well as the design of the business processes. Beyond this understanding, there are a number of other definitions of the term business model (cf., e.g., [14, 15]).

A business model therefore serves to understand the relationship between the company as a system of action and the creation of value. It reflects how a company works and what values it generates for specific target groups. Business models are created within the scope of a company foundation or a reorientation. They consist of several sub-models that describe which resources (materials, information, etc.) are (must be) available to a company as input variables, and how these resources are processed and transformed into marketable products or services, which are then transferred to the customer in order to generate corresponding revenues [16].

Business models can serve multiple stakeholders. The company management can thus better understand its own business and recognize existing strengths and weaknesses as well as opportunities for further development, transformation and improvement of its competitive position. For investors, the business model is often an important aspect of investment decisions.

A number of instruments have been developed to create business models. The best known is the Business Model Canvas by Alexander Osterwalder [17], which has found high acceptance in recent years. As the name suggests, the Business Model Canvas approach is based on a poster on which various aspects of the business model are visualized. The canvas provides a grid for nine business model aspects, which is filled with the concrete characteristics for a company at hand (see Fig. 2.3). The focus is on the value proposition (product or service).

When completing the form, a series of questions on each of the nine aspects must be answered. The following explanations briefly describe the aspects and provide a selection of associated questions.

1. Customer segments, target groups:
   All persons or organizations for whom the company in question wants to create values.
   Questions to be answered include:
   - Who benefits from the product or service?
   - Which customers are particularly important?
2. Value promise, customer benefit:
   Each customer segment has its own value proposition, the customer benefit. This is a combination of a product and service tailored to the needs of the respective segment.
   Questions to be answered include:
   - What benefit or value does the offer have for the customers?
   - Which customer problems are solved with the offered products and/or services?
3. Channels, sales channels:
   This factor represents the specific channels through which customers are addressed and promised values are communicated to them. Sales channels determine how interaction with customers takes place. Communication, distribution and points of sale form the interfaces between a company and its customers. The perception of the customer at these points of contact is central and determines the impression a customer has of a company.
   Questions to be answered include:
   - How do customers find out about the products and services offered?
   - How do the products/services reach the customer?
4. Customer relations:
   This section describes how dealings with customers are fostered.
   Every company should think about what types of customer relationships it wants to establish with different target groups. The design of customer relationships depends not only on the respective target group, but also on the associated objectives of the company (new customer acquisition, existing customer care, etc.).
   Questions to be answered are among others:
   - What kind of relationship do the individual customer groups expect?
   - How is the relationship with the customers organized?
   - How much does it cost to maintain customer contact and what is the value of this particular customer?
5. Revenue sources, revenue models:
   The company creates added value with its products and services. The central question is how much the customer is willing to pay for this. The company needs to decide on pricing models and pricing strategy (one-time payment, subscription, etc.).
   Questions to be answered include:
   - For what and how much are customers really willing to pay for the offer?
   - How much does each of the individual revenue sources contribute to total revenue?

- How would customers like to pay?

6. Key resources:

   Every company requires certain resources to prepare offers. These can be owned by the company itself, or also leased or provided by strategic partners.

   Questions to be answered include:

   - Which physical resources (facilities, production machines) are required to create and offer a product or service?
   - Which intellectual resources (knowledge, patents, partnerships, customer base) are needed?
   - Which personnel resources (teams) are required?
   - What financial resources (available capital, collateral) are required?
   - How can the necessary resources be procured and maintained?

7. Key activities:

   Key activities are the activities necessary for the creation and utilization of services, such as production, sales, and so on.

   Questions to be answered include:

   - Which key activities have to be carried out in order to offer a product or a service and thus realize the customer benefit?
   - Which activities are needed for which sales channels?
   - Which activities are required for which customer relationships?

8. Key partners:

   Key partners are business partners who provide important resources for the realization of the business model.

   Companies often enter into strategic alliances with these partners. Examples are suppliers, service providers, etc.

   Questions to be answered include:

   - Who are key partners and what do they do for the company?
   - Which key resources are provided by which partners?

9. Cost structure:

   The cost structure provides information on the most important cost factors of a business model.

   Questions to be answered include:

   - What are the largest and most important cost factors in the business model?
   - Which key resources/key activities are the most expensive?

Key Performance Indicators and associated target values for business processes can be derived from the individual parts of a business model. Conversely, the Key Performance Indicators and target values can influence the design of the processes. If the focus is on low prices, processes will look different compared to a business model focused on high quality.

Balanced Scorecard

The Balanced Scorecard (BSC) was introduced in the early 1990s by Kaplan and Norton [18]. It is a link between the business model, the development of a strategy, and its implementation. In the business world, strategy is classically understood as

the (usually long-term) planned behavior of companies in order to achieve their goals.

A BSC starts with the vision and strategy of a company and defines the Critical Success Factors (CSF) on this basis with the help of Key Performance Indicators and associated target values. The vision of a company describes the long-term ambitious goal that an organization or company strives for. Typical visions are formulations such as "we want to become the market leader in our market segment," or "we want to become the most profitable company in our market segment".

The Key Performance Indicators promote goal setting and performance in critical areas of the strategy to achieve the vision. The BSC is therefore a management system that is derived from the vision as part of the business model and the strategy to implement said model. It reflects the key aspects of the company. The BSC concept supports strategic planning and implementation by bundling the measures taken by all entities of a company on the basis of a common understanding of its goals and by facilitating access to the evaluation and updating of the strategy.

Since traditional management based purely on financial indicators no longer meets the requirements of companies for effective planning tools in the information age, Kaplan and Norton have introduced four perspectives for the BSC which allow the activities of a company to be assessed comprehensively. For each perspective, objectives, performance indicators, targets, and measures to be taken are defined (cf., Fig. 2.4).

Total Quality Management and EFQM

The term Total Quality Management (TQM) denotes the optimization of the quality of a company's products and services in all functional areas and at all levels through the participation of all employees. Optimization of quality means neither reaching the highest quality level with the given effort, nor increasing the quality without consideration of costs. Rather, it is a matter of focusing on the interests of the customer and determining quality in terms of the fulfillment of customer requirements.

The management of a company decides which requirements the company places on itself and which positioning toward the customer promises the most sustainable business success. This positioning is not static. Knowledge about customer needs and about the procedures to meet these needs require a continuous adaptation of the company.

In order to establish TQM, the European Foundation for Quality Management (EFQM) offers organizations assistance in setting up and continuously developing a comprehensive management system. Figure 2.5 shows the structure of the EFQM approach. On the one hand, this structure serves as a tool to build up a TQM and, on the other hand, to identify improvement potentials through a comprehensive evaluation system as well as to increase business success.

Enablers in the EFQM model are the methods and concepts used to achieve the results shown in the right half of the figure. The percentages in the presentation indicate the extent to which the individual aspects are included in the overall evaluation of the company.

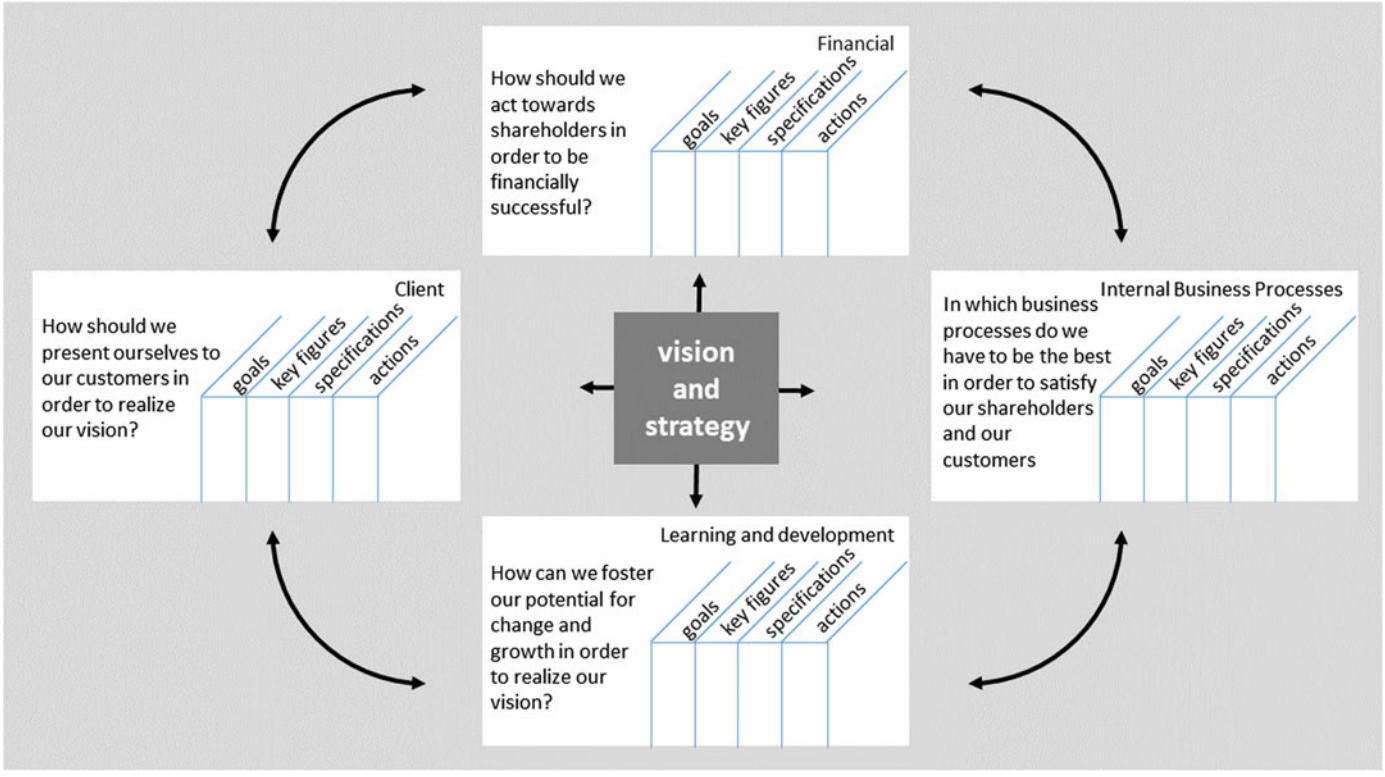

Figure 2.4: Perspectives of the Balanced Scorecard (adapted from Springer Gabler Verlag (publisher), Gabler Wirtschaftslexikon, keyword: Balanced Scorecard, online at: http://wirtschaftslexikon.gabler.de/Archiv/1856/balanced-scorecard-v8.html, last accessed 22.02.2018)

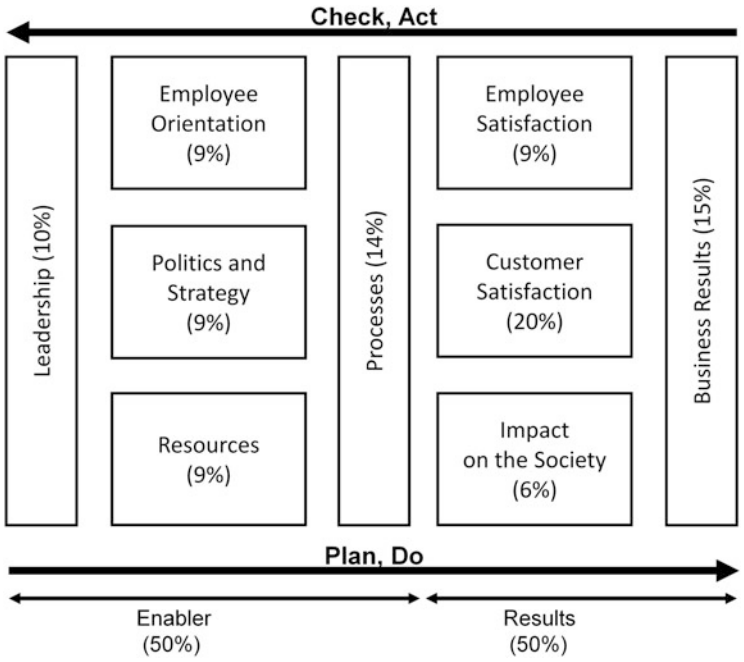

Figure 2.5: EFQM structure

The EFQM assumes that the enabling methods and concepts have the largest influence on the results (right side of Figure 2.5). The model thus provides good starting points for identifying Key Performance Indicators and their target values.

In addition to the possibility of setting up a management system, EFQM also offers a very sophisticated concept for evaluating its development status. A comprehensive catalogue of questions can be used to carry out an all-round evaluation of an organization. The evaluation can be carried out by employees of the organization itself or by external consultants. The best organizations in Europe score around 750 out of a maximum of 1,000 points in such evaluations.

EN ISO 9001

Compared to TQM, the EN ISO 9001 standard represents a weakened form of quality management. It describes minimum requirements for a quality management system. Figure 2.6 illustrates the basics of the standard.

Management's responsibility means that it defines which customer requirements are met and which quality policy is pursued. The implementation of the quality policy is planned, and the corresponding responsibilities and authorities are defined in the organization. Management is also responsible for evaluating the QM system at planned intervals and, in particular, taking customer feedback into account while doing so. Corporate management must also provide the necessary resources such as personnel, infrastructure, and an adequate work environment.

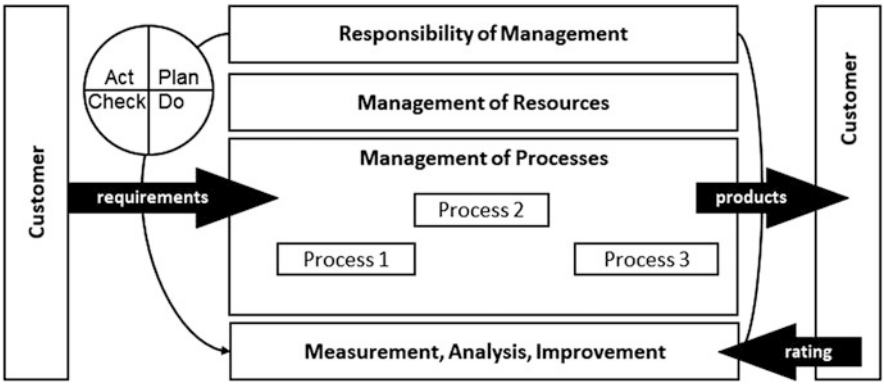

Figure 2.6: EN ISO 9001

The core of an EN ISO 9001-compliant QM system are the processes for realizing the products and the associated customer-related services. Tasks include the planning and definition of suitable processes for the development and manufacture of products, the procurement of inputs, etc. The tools used to monitor product manufacturing and quality must be regularly checked for their suitability. The execution of the processes must be continuously monitored through measurements and analyses of Key Performance Indicators, in order to be able to initiate appropriate improvement measures in the event of deviations.

EN ISO 9001 thus provides a framework for Business Process Management. The explanations show that, strictly speaking, there is no difference between Business Process Management and quality management. Without Business Process Management there is no quality management and vice versa.

The comparatively lower requirements of EN ISO 9001 are expressed in the fact that a company with a (merely) EN ISO 9001 compliant quality management system can only achieve about 300 points in an EFQM assessment.

Value Networks

The Value Networks concept was introduced by Verna Allee [19]. A Value Network is understood as roles and persons who exchange so-called tangibles and intangibles with each other. Tangible value flows are material value flows between roles and persons, and correspond to the exchange of goods, services, revenues etc. Tangible value flows represent transactions based on contracts. Intangible value flows are an additional benefit through the flow of knowledge; they are not contractually fixed or subject to a charge. Intangible value flows are, e.g., strategic information, planning knowledge, as well as existing emotional components such as mutual trust, common interests, need for knowledge, security, etc.

Value Networks should enable participants and organizational developers to actively shape social and professional relationships of interaction in organizational systems by visualizing and creatively handling mutual tangible and intangible performance flows (transactions). Figure 2.7 shows a simple Value Network. The

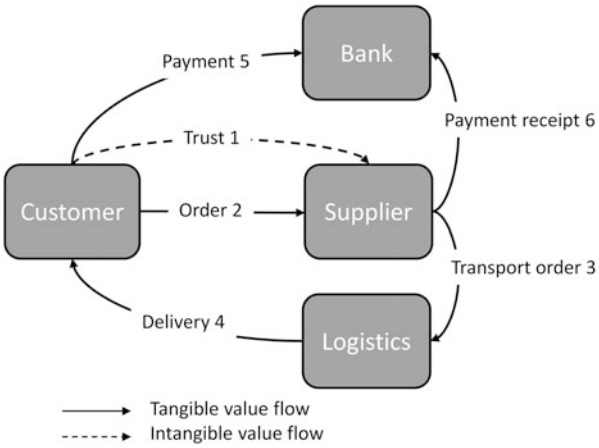

Figure 2.7: Schema of a Value Network

customer sends a value purchase order as a tangible value flow to the supplier. This tangible value flow is accompanied by the intangible value flow of trust. The customer and the logistics company are connected with the tangible value flow of delivery, etc. The numbers on the individual transactions express the sequence in which they are executed.

Organizations provide services as a result of their activities that ultimately contribute to the added value of a company or institution. In order to record these services and make them visible, an exchange-oriented view of organizations is recommended. This usually results in a network-like structure in which the roles within an organization and their interaction and communication channels are in the foreground. This perspective enables the transition to a communication-oriented Business Process Management.

2.5 Models of Business Informatics

Models in business informatics combine aspects from the economic and social fields with computer science to derive requirements for information systems. The models are mainly used to describe socio-technical human-machine systems. The social component covers the aspects around employees and partners. The technical dimension concerns the circumstances of Information Technology. It is important that corresponding models consider the interaction between the two domains, especially the human-machine interaction. In contrast to purely technical systems, which are regarded as deterministic, socio-technical systems can also be non-deterministic, i.e., complex, due to the involvement of social components. The state of development and research on the subject of modeling in business informatics can be found in [20]. We limit ourselves here to the handling of frameworks for enterprise architectures and IT service management.

In order to simplify the creation of very similar process models, reference models have been defined over the years by consulting firms or standardization bodies. A well-known example of such a reference model is ITIL (IT Infrastructure Library, [21]). Due to its wide distribution, we describe it in more detail as an example of a reference model. A reference model that includes ITIL while being more focused on governance and compliance, is COBIT [22]. Due to space limitations we do not provide a representation here but refer to the extensive literature and the official website [23].

Enterprise Architectures

The understanding of architecture in the context of companies coincides with the original meaning of the term architecture. In many areas of expertise it describes the basic organization of a system with its components and their relationships to one another and to their environment.

As already mentioned in Chapter 1, an enterprise architecture specifically describes and links the business and technical elements of an enterprise. The latter include in particular the IT landscape. Both the overall architecture and its parts are described through models. The range of model types used for this purpose extends from a business model through organigrams, data models and process models at the business level, to database models, algorithms and programs in the technical layer.

As for business models, there are also numerous frameworks for the modeling of enterprise architectures that provide orientation for those responsible and are intended to facilitate work processes. Dirk Matthes [24] has identified more than 40 frameworks with varying foci, levels of detail and degrees of familiarity. Only four of the most relevant will be handled in the following sections.

The Zachman Framework and The Open Group Architecture Framework (TOGAF) with its extension Architecture-Animate (ArchiMate) were named as essential frameworks in surveys (see also [24] page 5). The Architecture of Integrated Information Systems (ARIS) is widely used in practice in German-speaking countries and is of significant importance in the context of process management.

Zachman Framework

The framework presented by John A. Zachman in 1992 [25] in its extended form represents a structure grid similar to the Business Model Canvas, which the user has to complete with the facts for the enterprise at hand. It consists of a matrix with different perspectives in the rows and abstractions to each perspective in the columns. Figure 2.8 shows a condensed representation, a detailed picture can be found on the Zachman International website (www.zachman.com).

The perspectives in the rows have the following meaning:

- Planner: Company objectives, external requirements and influences, business model
- Owner: Requirements for data, processes, structures, etc. for company management
- Designer: System design and system structure to implement the requirements

	What	How	Where	Who	When	Why
Objective/area -> Role:Planer	List of important factors in the business	List of core processes, process map	List of branch ofices	List of mportant organizations and stakeholders	List of business events	List of business goals, strategies and business models
Business model -> Role: Owner	Conceptional data model/object model	Business process model	Business logistics systems, distri-bution model	Workflow model, organizational structure	Schedule	Business plan
System model (logical) ->Role: Designer	Logical data model	Process implementation, system archi-tecture model	Distributed systems architecture	Human interface architecture role model	Process structure	Busines rules model
Technology model (Physically) -> Role: Builder	Physical data model, class model	Technology design model, implementation model	Technology architecture	Representation architecture, system usage	Control structure, system dynamics	System requirements
Detailed presentation (from context) -> Role: Programmer	Data definitions, database structure	Program, process configuration	Network architecture	Security architecture, directory services	Schedule	System monitoring
Operations ->Role: User	Concretly available equipment	Process instances	Business locations	Execution	Current schedules	Current moods
	Inventory, material and data	Processes and functions	Spatial distribution, geometry	People and machines	Scheduling	Goals, motivation

Figure 2.8: Zachman framework

- Builder: Implementation of the system design
- Programmer: Provision of the technical infrastructure
- User: responsible person for the operation to ensure the functionality

The columns contain the questions that the company needs to answer:

- What (inventory): What objects, equipment, data, information, etc. are required?
- How (functions and processes): How does the company work, for example, what do the business processes look like?
- Where (locations, network); Where are the company's locations?
- Who (people): Who are the people who keep the company running? Which business units are there and what is the organizational structure like?
- When (Time): When are business processes instantiated and executed? What are the time schedules for the business?
- Why (motivation): Why do we run the business the way we run it? What are the drivers of the business? Aspects of the business model are incorporated here.

Zachman envisages that a suitable model will be developed for each cell in the table. From this perspective, his framework is a model for a set of models that allow a closer look at different aspects of a company.

The users can deviate from the original structure in the rows and columns by changing the emphasis. This flexibility is a strength of the model frame. However, it does not contain any procedure or methodology for defining a concrete enterprise architecture. Processes for their development or transformation have to be exploited elsewhere by the users or have to be designed entirely by themselves.

The Open Group Architecture Framework (TOGAF)

TOGAF is the Open Group's framework for the development of enterprise architectures, including business processes [26]. While the Zachman framework emphasizes the object perspective and offers little support for the architecture development process, TOGAF focuses on the procedure for model creation. It provides methods and tools that help with the introduction, creation, use and further development of enterprise architectures.

TOGAF distinguishes four sub-architectures:

- Business Architecture: Business aspects of enterprise architecture
- Data Architecture: Logical and physical structures of data and resources for their management.
- Application Architecture: The application systems used and their relationships with each other as well as their relevance for the company's business.
- Technology Architecture: Software and hardware requirements for data management and application system execution. This includes, e.g., runtime environments, networks, middleware, and other operational infrastructures.

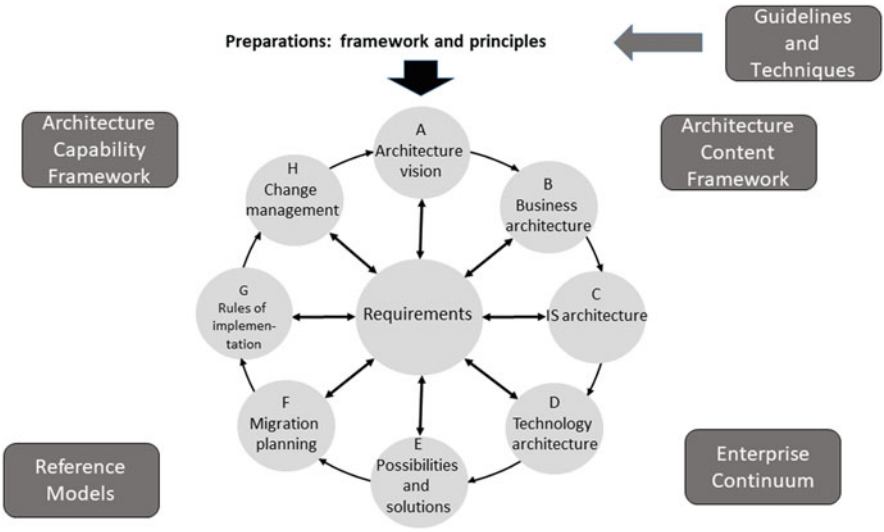

Figure 2.9: Architecture Development Method from TOGAF

The TOGAF framework consists of the following components:

- Architecture Development Method (ADM)
  Method and procedure for the development of an enterprise architecture.
- ADM Guidelines and Techniques
  Set of tools and guidelines that support the use of ADM (e.g., tools for iterative use of ADM).
- Architecture Content Framework
  Structural model to define, structure and display the results generated with ADM in a uniform and consistent way.
- Enterprise Continuum
  Model for structuring a possible repository that can contain the respective architectures and the possible solutions such as models, patterns, architecture descriptions, etc.
- Reference Models
  Basic models that can be used as a basis for specific models for a company. These are the Technical Reference Model (TRM) and the Integrated Information Infrastructure Model.
- Architecture Capability Framework

Various reference materials for the development of specific architectural models. The Architecture Development Method (ADM) forms the core of TOGAF as an iterative process model (cf., Fig. 2.9). This creates all of the architecture artifacts. ADM can be applied at multiple levels, allowing architects to define different levels of detail of the enterprise architecture. With the help of the other components, the results are then described, structured and stored.

The phases of the ADM are:

- Preliminary phase: Here, the organizational environment and the frameworks, methods, support tools and important principles used are defined.
- Phase A - Architecture Vision: Here, the goals and the parties involved in updating the enterprise architecture are defined and integrated.
- Phase B - Business Architecture: The current and desired state of the business architecture is described here. The decisive differences are worked out. The desired views are defined and the associated appropriate tools are selected.
- Phase C - IS architecture (Information System Architecture): The current and desired state of the application and information/data architecture is described here. The decisive differences are worked out. The concrete applications and data models are used for this purpose.
- Phase D - Technology Architecture: The current and desired state of the technology architecture is described here. The decisive differences are worked out. In addition, the concrete hardware systems are described.
- Phase E - Opportunities and Solutions: Here, the projects are defined which carry out the transformation from the current situation to the target state.
- Phase F - Migration Planning: The transfer from a current state to a target state is planned here.
- Phase G - Implementation Governance: The implementation into the target state is carried out and monitored here.
- Phase H - Architecture Change Management: Requirements and external influences are collected here, which then serve as the basis for the next ADM run.
- Requirements management: The requirements management drives the ADM process continuously and is therefore at the center.

Architecture-Animates [27] (ArchiMate)
Architecture-Animate (ArchiMate) is the name of an open and independent modeling language for enterprise architectures published by the Open Group. It provides tools that enable enterprise architects to describe, analyze, and visualize the relationships between business units and their development.

The ArchiMate language enables the description of the structure and flow of business processes, organizational structures, information flows, IT systems, and technical infrastructure. The descriptions help the participants to design changes in architectural elements and their relationships, to evaluate the consequences and to communicate them. Figure 2.10 shows the ArchiMate framework.

The first three columns correspond to the basic concepts of ArchiMate:

- Passive structure elements

  Passive structure elements are the objects on which the actions from the behavior (behavior elements) are executed. In general, these are information objects, but physical objects can also be modeled as passive structural elements.

	Passive structure	Behavior	Active structure	Motivation
Strategy	Resources		Resources	Stakeholders, drivers, goals, principles and requirements
Business	Business objects	Business services, functions and processes	Business actors and roles	
Application	Data objects	Application services, functions and processes	Application components and interfaces	
Technology	Artifacts	Technology services, functions and processes	Devices, system software, communication networks	
Physical	Material			
Implementation and migration	Deliverables	Work packages	Platforms	

Figure 2.10: Elements of the ArchiMate framework

- Active structure elements
  Active structure elements are elements that can perform actions. Examples are people, applications, computer nodes, etc. The actions can be triggered via interfaces, which also provide the results.
- Behavior elements
  Behavior elements represent the dynamic aspects of a company. A service is the externally visible behavior of the system that provides the service. The services are used via the corresponding interfaces. Interface events trigger the active structure elements, which then execute the corresponding service function.

These three model fragments correspond to the basic elements of natural languages: subject, predicate or verb and object. They are considered on a total of six layers:

- Strategy layer
  Motivation describes what a company wants to achieve. The strategy concepts describe at a high level of abstraction how a company wants to achieve its goals.
- Business layer
  The elements of the business layer can be used to describe products and services that a company makes available externally. The business layer shows how the company realizes these products and services and is intended to help with the analysis of the corporate structure.
- Application layer
  In the application layer, the support of the business layer is represented by applications and data.
- Technology layer
  In the technology layer, the infrastructure needed to implement applications is described. These are essentially the required hardware and software components.
- Physical layer
  This layer focuses on the interaction of IT and physical components such as machines, sensors and actors.

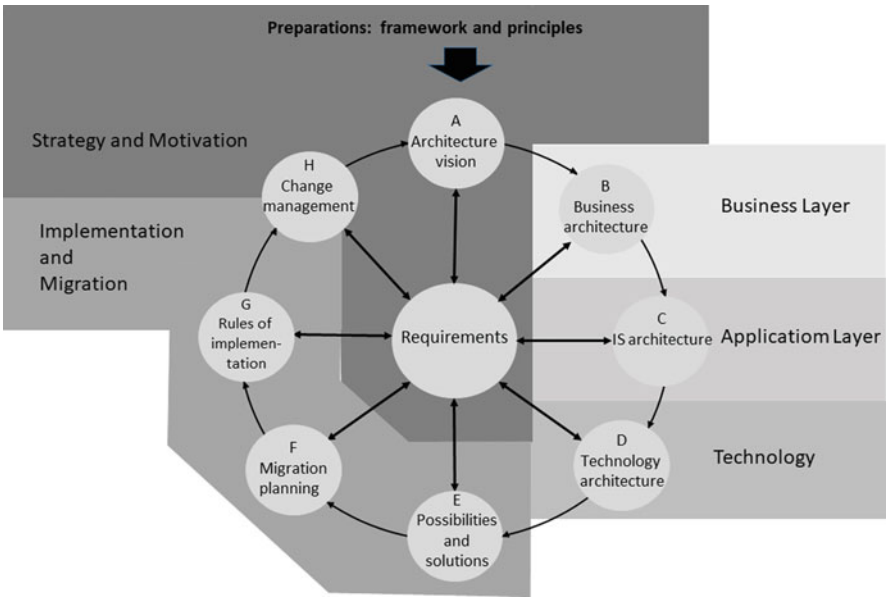

Figure 2.11: Relationship between ArchiMate layers and TOGAF ADM

- Implementation and migration layer
  The implementation and migration concept describes how a defined architecture is to be implemented. In particular, it describes the work packages for implementation.

The cells of the table therefore contain the core elements for the active and passive structures as well as for the behavior in the respective layer.

The motivation column describes the reasons for designing or changing an enterprise model. These influence the modeling and give it the appropriate direction.

ArchiMate is seen as a supplement and concretization of TOGAF. TOGAF describes the process for the definition and description of an enterprise architecture (enterprise model), but it does not contain any description languages for the respective sub-models. ArchiMate aims to fill that gap. In addition to the framework shown for the architecture to be developed, it also offers languages for expressing the aspects in the individual layers. Figure 2.11 shows how the individual steps in TOGAF's architecture development method are related to the ArchiMate layers.

Architecture of integrated information systems (ARIS)

The Architecture of Integrated Information Systems (ARIS) is a framework for the definition of enterprise models. It includes the data, function, organization, control, and performance views. For each view, ARIS provides a number of model types for documentation.

- Data view:
  The data view comprises the business-relevant business or information objects and their relationships to each other, that is, all data that is related to the activities of a company. Information objects include states such as article or customer status as well as events such as "Sales order has arrived" or "Production order has been triggered". The relevant model type is the Entity-Relationship Model (ERM).
- Function view:
  The function view describes the business-relevant activities (functions, activities) and their hierarchical relationships. Subordinate functions are sub-functions of the higher-level function. The functions perform operations on the objects described in the data view. In practice, functions are modeled with function trees.
- Organizational view:
  In the organizational view, the organizational structure, that is, the personnel resources of a company and their hierarchical relationships are modeled in an organizational plan. The organigram is the usual model type to represent the organizational view.
- Control view:
  The control view establishes the chronological and factual connection between the individual operational activities. It merges the data, functional and organizational views and thus plays a central integrating role. The control layer is therefore also called the process view. The main model type is the Event-driven Process Chain (EPC).
- Performance view:
  In the performance view, the entries and results of the business process at hand are usually described using product trees.

The views, typical model types and their context are shown in Figure 2.12. The picture also makes clear that ARIS, with its integrated control view, places business processes, especially the sequence of activities, at the center of the consideration.

Orthogonal to the structuring carried out by the views, ARIS differentiates the abstraction levels functional concept, data processing concept and implementation on the basis of software engineering. This shows the close relationship between the models developed with ARIS and Information Technology. To solve an operational problem, a functional model is created and transferred to a corresponding data processing concept (data processing model). This ultimately serves as the basis for the concrete technical implementation.

- Functional concept:
  The functional concept describes the facts of the operational problem. At this level, data models, functional models, organizational charts, value chains or Event-driven Process Chains (EPCs) and product models are used.
- Data processing concept:
  The data processing concept specifies how the functional concept is to be implemented in IT terms. At this level, database models (data view), structure

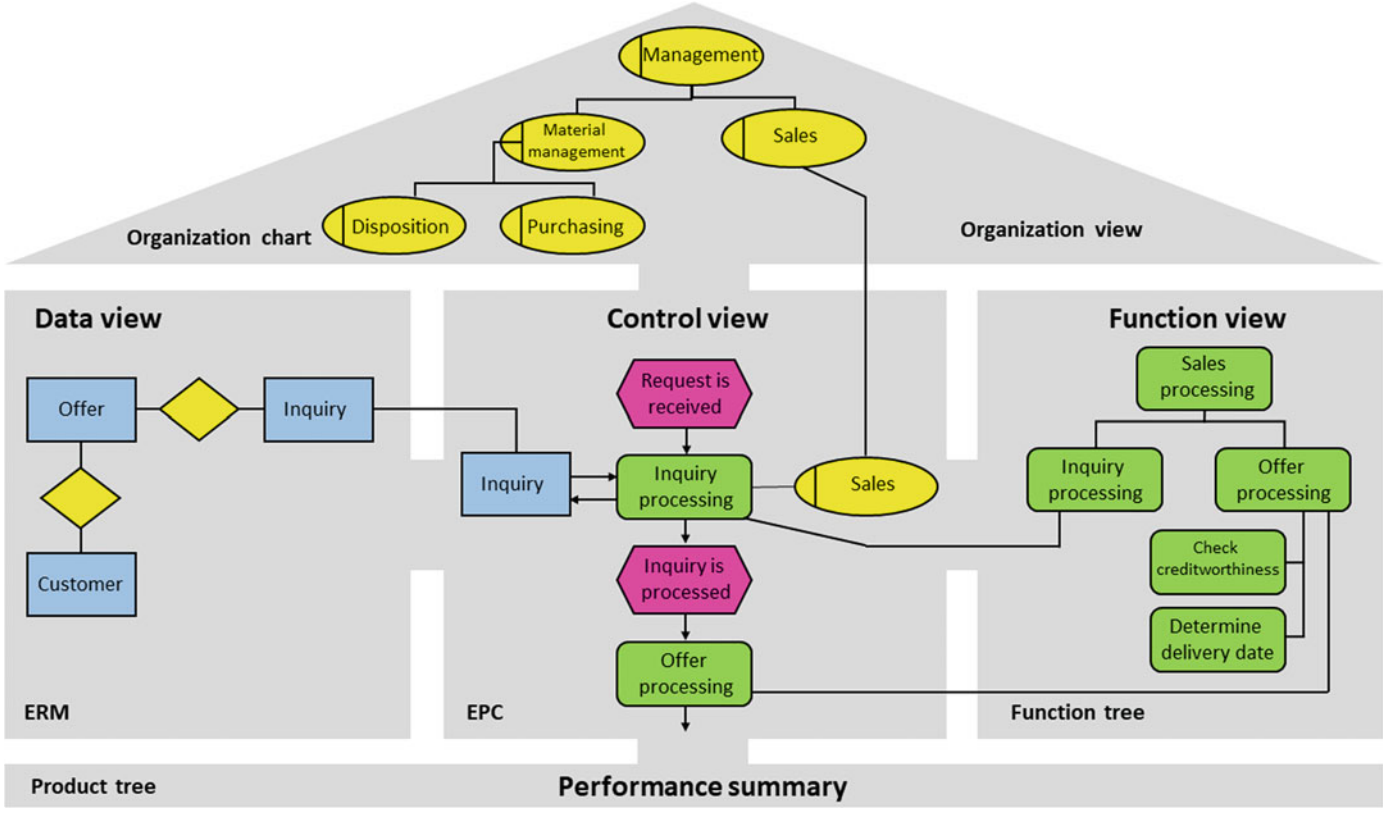

Figure 2.12: Views and relevant model types in ARIS

charts (functional view), network topologies (organizational view) and trigger mechanisms (control view) are considered. The purpose of the data processing concept is to adapt the functional concept to the requirements of Information Technology.

- Implementation:

  At this level, the data processing concept is converted into an executable software system. At this abstraction level, data description languages (data view), programs (function view), network protocols (organizational view), and program control (control view) are considered.

Framework for IT Service Management: ITIL

The IT Infrastructure Library (ITIL) is a collection of predefined processes, functions and roles that typically occur in every IT infrastructure of medium-sized and large companies [21]. The practical assignment of activities is based on roles and functions. These are best practice proposals that have to be adapted to the needs of the company. The collection has since been supplemented by ISO 20000:2005, an ITIL-based certification model for organizations.

The IT Infrastructure Library comprises five core volumes with a current total of 37 core processes. Figure 2.13 shows the structure of ITIL. The five core volumes are based on the service life cycle. Based on the service strategy containing the processes strategy development, financial management, service portfolio management and demand management, the service is finally provided via the process groups of service development and service commissioning with the processes of the service provision group. The procedures are subject to continuous improvement.

Further books such as "Software Asset Management", "Small-Scale Implementation" or "Building an ITIL based Service Management Department" supplement the core publications. ITIL thus offers comprehensive support in the development of a process system for the IT department of an organization.

2.6 Models in Computer Science

Models in computer science relate to data structure models and processes in computer systems as well as to various essential accompanying aspects such as security, robustness, etc.

On the one hand, these models serve to illustrate a considered part of reality in order to solve a task with the help of information processing. They refer to a defined problem area or certain application areas of computer systems. This includes models that focus on data and the operations running on it, as well as models that look more at the overall structure of complex programs, i.e., their architecture. These two model categories are also called models for programming on a small scale or models for programming on a large scale respectively.

In addition to these central model categories of computer science, there are other models that consider flanking aspects such as access models, security models, etc. These are not the subject of further discussion here.

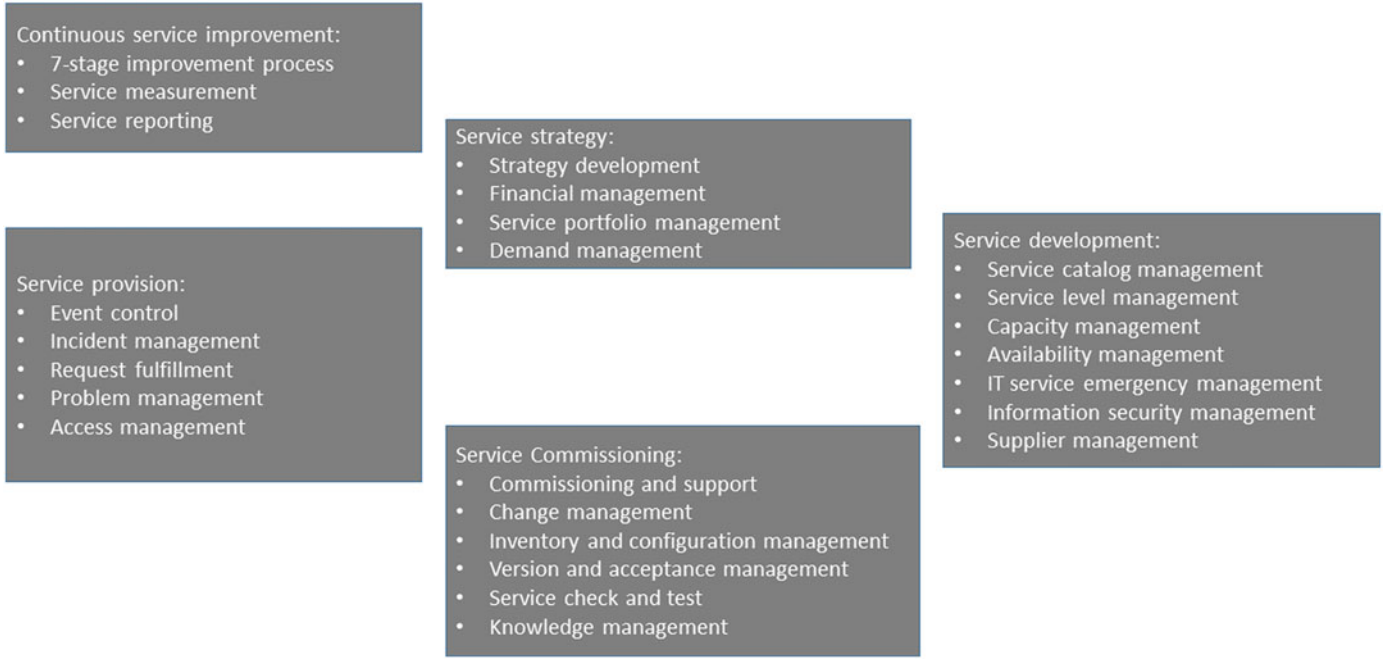

Figure 2.13: ITIL process groups

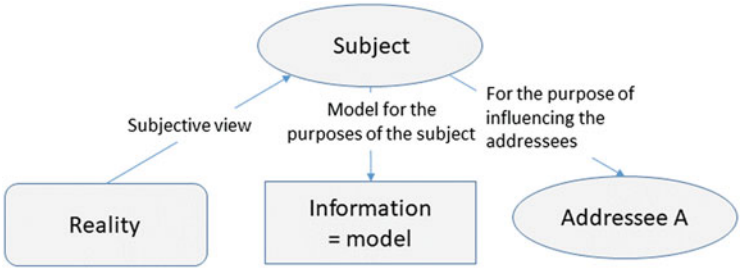

Figure 2.14: Information is “model - from what - for what - for whom" [11]

Instead, in the following sections we explain several model concepts considered essential for Business Process Management.

Information

Information is the more central aspect of data processing, which is why it is also called Information Technology or IT. Information and information systems are models that represent an object of the real world according to the ideas of one or more subjects. The subject's ideas are oriented toward the intended purpose. For example, items in a warehouse are described by their properties such as part number, dimensions, weight, etc. Different users use different parts of the model: The purchasing department looks at information such as purchase price and order limit, while the warehouse worker is more interested in dimensions and storage location.

The modeling of the object reality can therefore be understood as an interpretation by the user agents (subjects) such as purchasing or warehouse clerk. Information is then the result and the reason for an interpretation, but it can also itself be an object and thus an object of interpretation and modeling. This relationship between subject, information (model) and original is shown in Figure 2.14.

Information receives its value through the interpretation of the overall event by the observing subjects. This observation is partly conscious, but mostly unconscious. The amount of information is reduced and filtered according to the respective need for knowledge, or linked with other information.

Data is different from information. A data element is initially only a sequence of characters whose meaning is not unique. The characters can be numbers, letters or symbols. In the marketing department of an online shop, for example, the sequence of numbers 0815 may be found. Although this sequence of characters represents a date, its meaning is not known. The string itself has no meaning except for its individual elements.

From this data, however, information can arise, if it is known in which context it is to be interpreted. By combining it with other data, a relationship is created that can be interpreted, and information can be generated. If the data 0815 is in the context "Customer Max Sample, Article 0815", the marketing department can interpret that the customer Max Sample ordered an article with the number 0815. The supplementation of data with other data depends on a subject's interest in knowledge or his

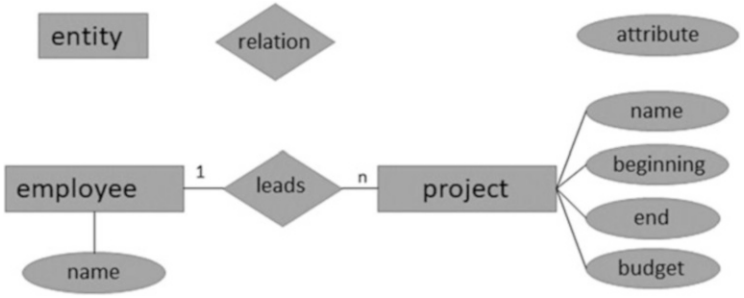

Figure 2.15: Example of an Entity-Relationship Diagram

intentions to use it. With the information produced in each case, a subject usually wants to influence an addressee, e.g., to perform an action. Information is thus to be understood as "statements that improve the degree of knowledge of a subject (information subject/user) about an object (information object) in a given situation and environment (information environment) in order to fulfil a task (information purpose)" [28] (cf., Fig. 2.14). Modeling is therefore a part of information management.

Information is important for politicians and business leaders, but also for every citizen of the world. It reflects a particular situation that applied at a particular point in time and usually allows an update into the future. It serves to make political, economic or personal decisions. When using information, the question always arises of who created it and what intentions that person has. The subjectivity of models thus plays an essential role here.

Entity-Relationship Model

In order to turn data into information, it must be combined with other data and the relevant relationships described. This creates a data model. The best-known method for describing data models is the Entity-Relationship Model (ERM). An Entity-Relationship Model consists of three main elements:

- Entities
  Entities are the object classes that are considered in the part of the real world that is of interest.
- Relationships or relations
  Relations describe the relationships between entities.
- Attributes
  Attributes are properties within the context of an entity.

Figure 2.15 shows an example of an ERM. It also indicates which symbols are generally used to express the main elements. However, one can also find ERM representations with notation elements of the Unified Modeling Language (UML).

Figure 2.15 describes the following situation:

An employee has a name. A project has a name, a start, an end and a budget. The so-called cardinality expresses the fact that an employee can lead several projects, but a project can only be led by exactly one employee.

In the modeling concept of classification, objects are combined to form object types (entity sets) and relationships are combined to form relationship types (relationship sets).

These types are differentiated according to:

- Entity type: Typecasting of similar entities (e.g., employee, project)
- Relationship type: Typecasting of similar relationships (e.g., employee leads project)
- Attribute type: Typecasting of similar properties (for example, name for the entity type employee). Attributes or combinations of attributes whose value(s) uniquely identify an entity are called identifying attribute(s) (e.g., the attribute project name identifies the entity type project).

Flowcharts

Flowcharts illustrate an execution sequence of activities or actions and are used in numerous application areas. They can describe the order in which actions are to be performed by people or other actors. Algorithms or computer programs are often documented in the form of flowcharts (e.g., program flowcharts). Due to the broad application of flowcharts, numerous variants have developed which take into account special circumstances of the respective field of application. For data processing, the symbolism for flowcharts was defined in the standards DIN 66001:1983-12 and ISO 5807:1985. Figure 2.16 shows an example of a flowchart.

In the explanation of ARIS in section 1.5, we addressed Event-driven Process Chains (EPC) as a model type for the control view. These EPCs are flowcharts of sequences of event nodes, function nodes (operations), and connectors. Arrows as edges represent control flows between the symbols. Functions and events (with the exception of start and end events) each have exactly one incoming and one outgoing edge. If functions are to create several events or if several events should trigger a function, connectors such as an exclusive or (XOR), must be used. The modeler can also express who executes a function with which IT support and what data is manipulated in doing so. For this purpose he assigns symbols for organizational units (for example, departments, jobs, roles), information objects (data), or application systems to the function nodes. These elements must be specified in the corresponding model types. This is referred to as extended Event-driven Process Chains (eEPC).

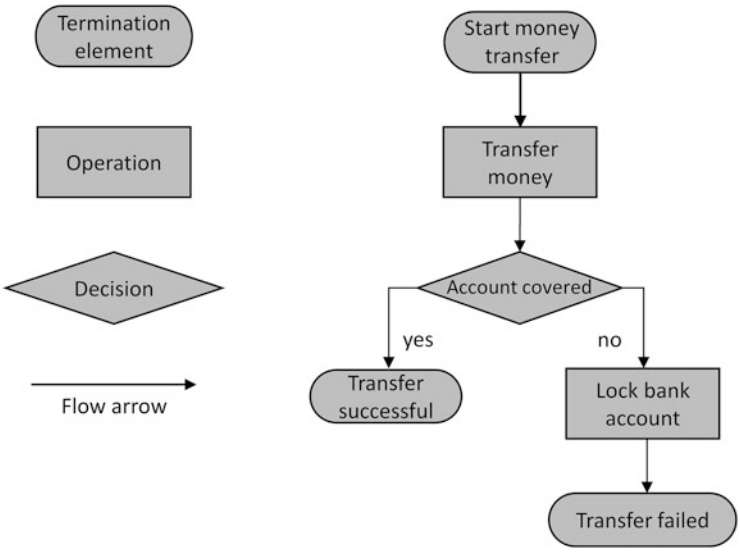

Figure 2.16: Flowchart for account transaction

We have already briefly indicated in section 1.5 that an eEPC, as an instrument to describe the control or process view; correlates the interaction between the elements of the other views and model types. Specifically, it expresses itself in the following way:

- Control view
  - An event is a state that occurs before or after a function. The symbol for an event is hexagonal.
  - A function (process) is an action or task that follows an event. Functions are symbolized by rectangles with rounded corners.
  - Connectors are used to split or join the control flow. The three connectors AND, OR and XOR are available, each represented in a small circle with the corresponding symbol. The decision as to which path is followed after a connector is made by the function preceding the connector.
- Functional view
  - The function nodes in the control view are linked with nodes from the function tree of the function view and thus specify the activity to be executed.
- Data view
  - Information objects are entities from the data model that are bound to carriers such as documents or other data stores. They represent inputs or outputs of the function to which they are connected by a directed edge. The symbol for an information object is a rectangle, the character as input or output is determined by the arrow direction of the connection edge.
- Organizational view
  - Organizational units show which elements from the organizational chart, modeled according to the organizational view, execute the activities (functions) in the process. Organizational units are connected to functions by undirected edges.

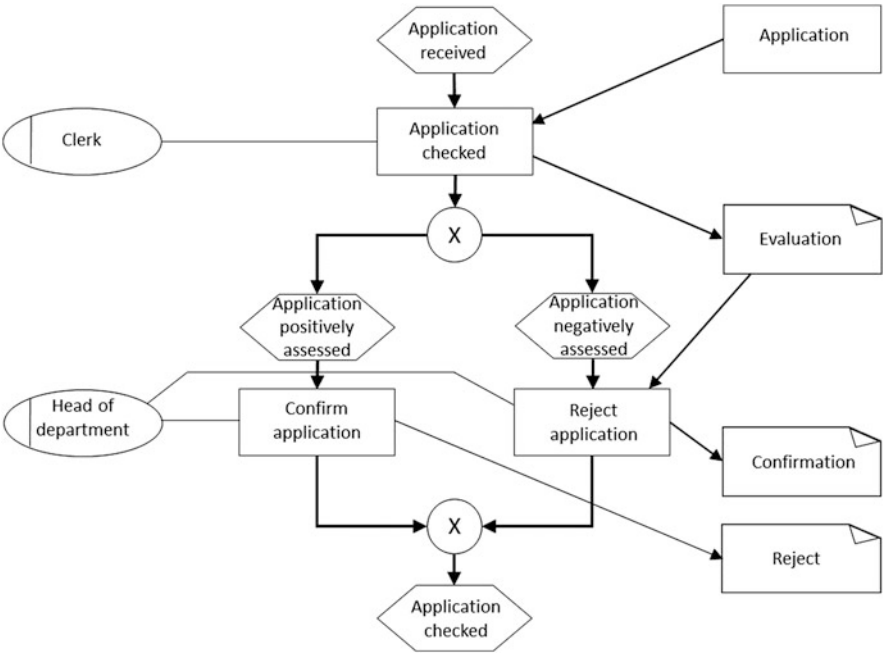

Figure 2.17: Extended Event-driven Process Chain for an account transaction

Figure 2.17 shows an example for an extended Event-driven Process Chain (eEPC).

Petri Nets [18]

One of the first and most widespread theoretical approaches to the description of parallelism are Petri nets.

Petri nets are used for the logical modeling of behavior. They consider the behavior of systems, usually information systems, under the following aspects:

- Activities to be carried out
- Preconditions and postconditions of an activity
- States of all conditions (possible values for each individual precondition or postcondition): The state of a condition is the distribution of so-called tokens as a state display to the pre- or post-range.
- Initial state (initial token)
- Procedures (possible consequences of activities)

A Petri net is a structure that is formally and mathematically precisely described as a directed graph with nodes consisting of two disjoint subsets marked with tokens. In the representation as quadruple the following applies: A Petri net is a quadruple: PN = (S, T, K, M) with

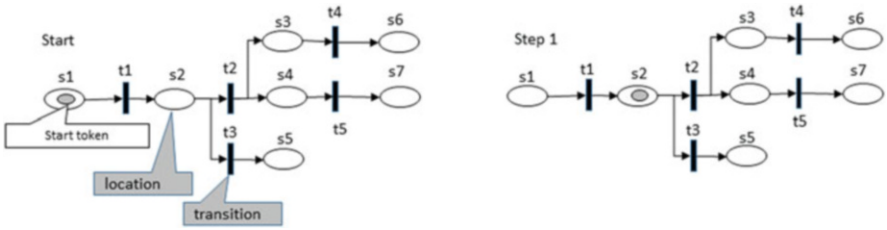

Figure 2.18: Petri net I

- s ε S: Places (i.e., conditions), for the description of states and/or conditions, buffers, memories or storage spaces. They are circular in the graph and are used to store information or tokens.
- t ε T: Transitions, describe state transitions, events, actions or activities and are shown in the graph as line, bar or cuboid forms. Their purpose is the processing of information.
- k ε K: Arcs are possibly weighted (i.e., numbered) connections between places and transitions, shown in the graph as arrows. They indicate the course of the transitions.
- m ε M: Tokens that represent the current state of the Petri net. Every network has an initial state, i.e., there is an initial token.

According to the above definition, arcs only go from place to transition or from transition to place. Input places of a transition t are places from which arcs run to the transition t. Output places of a transition t are places to which arcs of the transition t lead.

A transition can switch if there is at least one token in each input place. Switching means that a token is removed from each input place of the switching transition and a token is added to each output place.

The following figure 2.19 show a Petri net with its respective states after performing the switching operations. In the start state in Figure 2.18, only the place s1 has a token, the initial token or the start token. The place s1 is the only input place of transition t1, which can therefore switch. The token is removed from s1 and a token is added to the single exit place s2. The right half of the following figure 2.19 shows the state after switching t1.

The place s2 is the input place of the two transitions t2 and t3 (see left side of Fig. 2.19). So it could switch the transition t2 or t3, but not both. Which transition switches is random. This gives us a so-called non-deterministic state. If t3 switches, a token is added to the place s5 and a final state is reached, since no other arc leads out of s5. If the transition t2 switches, a token is added to the places s3 and s4, which are starting points of t2. The corresponding state is shown in the right half of Figure 2.19.

Now either the transition t4 or t5 can switch. Here one could conclude that these two transitions switch simultaneously. But this is not allowed in Petri nets. Only one

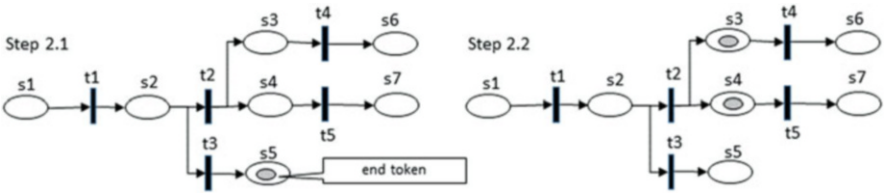

Figure 2.19: Petri net II

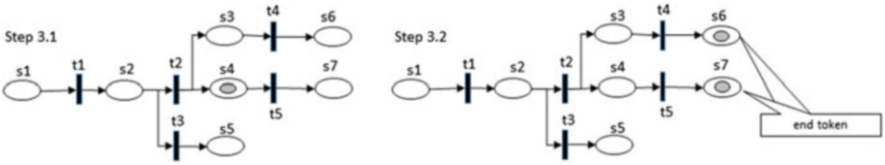

Figure 2.20: Petri net III

transition may switch at a time. Consequently, parallelism as observed in reality cannot be represented. However, it is arbitrary whether t4 or t5 switches first. In our example, t4 switches first and then t5. When both transitions have switched, a final state is reached again. Figure 2.20 shows the corresponding network states.

Petri net models allow the analysis and simulation of dynamic systems with concurrent and non-deterministic processes.

The type of Petri nets presented here is the basic version. It cannot be used to describe certain situations. For example, it is not possible to assign priorities for transitions, which is why, for example, the right-before-left rule at traffic junctions cannot be modeled.

In order to remedy such deficits, extensions have been introduced to map further aspects of reality to the model or to describe certain situations more compactly. Examples are multi-valued, colored or prioritized Petri nets. Details of such extensions can be found in the available literature, e.g., [29].

Calculus of Communicating Systems

The Calculus of Communicating Systems (CCS) was published by Robin Milner in 1980 (see [30]). This calculus enables the formal modeling of parallel communicating systems. This permits networked systems with a static topology to be described. CCS can be used to formally investigate the properties of programs such as deadlocks, bisimilarity, etc. CCS enables the description of the following aspects:

- Communication between actors via channels
- Interaction with the environment, i.e., reactivity
- Parallel composition
- Hiding actions from the environment (information hiding)
- Non-deterministic branches

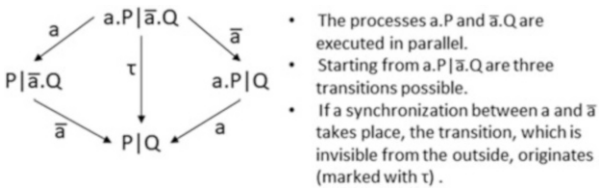
a.P|ā.Q
a
ā
P|ā.Q
τ
a.P|Q
ā
a
P|Q
• The processes a.P and ā.Q are executed in parallel.
• Starting from a.P|ā.Q are three transitions possible.
• If a synchronization between a and ā takes place, the transition, which is invisible from the outside, originates (marked with τ).

Figure 2.21: Process interaction with CCS

The course of a process is described as a tree, i.e., there is a root that represents the initial state and from where the individual branches originate. Each of the branches is marked. These markers represent the actions performed to move from one state to the next. A distinction is made between observable and unobservable actions. Unobservable actions can be executed at any time within a process without affecting other processes. Processes have no common variables.

Recursive expressions are used to describe the behavior of a process. Within behavior expressions, variables can be used to reference other behavior expressions.

The behavior expressions are described according to the following syntax, in which uppercase letters denote process names and lowercase letters denote actions.

- Empty process:
  Ø
- action
  Process a.P1 executes action a and then behaves like P1.
- process name
  With the expression A := P1 the process P1 gets the name A. Since recursive definitions are allowed, the expression P1 can contain the name A again.
- choice
  Process P1+P2 can be continued with either process P1 or process P2.
- parallel composition
  P1|P2 means that processes P1 and P2 are executed in parallel.
- renaming
  P1[b/a] describes the process P1, in which all actions with the denotation a are renamed to b.
- restriction
  P1\a denotes the process P1 without the action a.

Matching input and output actions in two different processes can synchronize and become an internal action τ. In general, the coaction for an action is marked with a line above the action name. These complementary actions are the send and receive actions. Figure 2.21 shows an example of the interaction between two processes.

The example in Figure 2.22 shows how a simple holiday request process can be described with CCS.

In the employee process, the vacation request send action is executed (action name with overline). The system then waits for the messages 'rejected' or 'approved'. The manager receives the vacation request message, which he replies to either with

Employee := $\overline{\text{vacation request}}$. (Rejected + Approved) . NIL

Manager := vacation request. $\overline{\text{(Rejected}}$ + $\overline{\text{Approved}}$.$\overline{\text{approved vacation request}}$).NIL

Travel departement:= approved vacation request.NIL

Vacation application process := Employee | Manager | Travel departement

Figure 2.22 Example process in CCS

the message 'approved' or 'rejected'. If he sends the message 'approved', the message 'approved vacation request' is also sent afterwards. This message is received by the travel department. Messages are exchanged synchronously in CCS, that is, a sender waits until the receiver executes the corresponding receive action.

There are also formal rules of derivation for the informal interpretation. This makes process definitions accessible for formal evaluations.

π Calculus

CCS only allows static process structures. Communication relationships cannot be changed dynamically. The π Calculus (see [31]), also developed by Robin Milner, allows the representation of processes with changing structures. Any connections between components can be displayed and these connections can also change, or new ones can be created. Thus, the π Calculus is an extension of CCS to include concurrency. The notation in the π Calculus is largely based on the CCS notation. The following example explains the modeling possibilities of the π Calculus (see Figure 2.23).

The agent (process) P wants to send the value 7 to Q via a link a. However, the value is to be transmitted indirectly via another agent R.

In Figure 2.24 the individual steps for the execution of system O are shown.

The processes P, R and Q are executed in parallel. Process P sends via channel b the name a and then the name 7. Process P receives via channel b the two names. This means that each x is replaced by a and each z by 7. In Figure 2.24 this is the result after step 2. Now the name 7 can be sent via channel a, which is then accepted by process Q. Thus, the value 7 was sent to process Q via process R. The following graphic shows how the structure of System O is changed by its process flow.

Since the channel name a is transmitted from P to R, the processes R and Q are linked via channel a after the message has been accepted. This property shows that the π Calculus, in contrast to CCS, permits the modeling of structural changes (cf., Figure 2.25).

Communicating Sequential Processes

Communicating Sequential Processes (CSP) is a methodology for describing interaction between communicating processes. The idea was first introduced by Tony Hoare in 1978 as an imperative language (see [32]). It was then developed into a formal algebra and made famous in 1985 with the publication of the book Communicating Sequential Processes (see [33]).

- P:= $\bar{b}a$. $\bar{b}7$.P' Process P transmits the value a via channel b and subsequently the value 7 and then behaves like P'.
- R:=b(x).b(z).$x\bar{z}$.0 Process R receives any value via channel b. This means in R, x and z are replaced by the received values.
- Q:=a(x).Q Process Q receives via channel a any value for x. The received value is used everywhere where x is located.
- O:=P | R | Q Entire system O

Figure 2.23: Process description with π Calculus

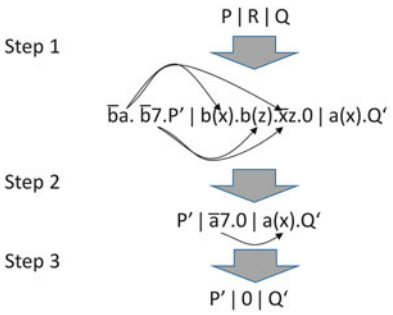

Figure 2.24: Execution steps of system O

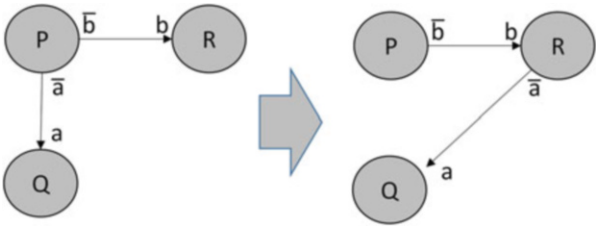

Figure 2.25: Structural change

In CSP, as in CCS, the number of processes is static. It cannot be changed during runtime and there are no common variables between processes. Instead, the processes 'know' each other and communicate with each other by sending and receiving messages. For sending, the send process P executes the output command Q! (expr) and the receiver process Q receives the input command P? (vars). Output and input commands are called corresponding if the sequence of expressions (expr) and the sequence of variables (vars) are of the same type in relation to their numbers and components. Analogous to CCS and the π Calculus, CSP is based on an unbuffered

message exchange in which the send and receive processes must be explicitly named.

With Q!() and P?() a message without content is sent. Such messages are called signals and only serve to synchronize processes. If different signals are required, the distinction is made by means of type designators of the form Q!(Signal1) and P?(Signal1).

In addition to the unconditional message exchange described above, there is the receive instruction within a so-called guarded command. A guarded command is only executed if the preceding Boolean condition is true. The formula set

$$x > y; P?(z) -> x := x + y; y := z$$

is only executed if x is greater than y. Then the message is received by P if P is ready to send the message.

To be able to wait for messages from different senders, several guarded commands are combined to form an alternative instruction.

$$[\, x > y; P?(z) -> x := x + y; y := z \,||\, x < y; Q?(z) -> y := x + y; y := z \,]$$

In case $x > y$ the message P?(z) is expected, in case $x < y$ the message Q?(z). Alternative instructions can also be executed repeatedly. Syntactically, this is expressed by a $*$ in front of the alternative instruction.

$$*[\, x > y; P?(z) -> x := x + y; y := z \,||\, x < y; Q?(z) -> y := x + y; y := z \,]$$

The instruction is executed until none of the conditions are true, then the repetition is terminated.

The concepts regarding the concurrency of CSP serve as a design basis for the programming language Go.

Abstract State Machines

In computer science, an Abstract State Machine (ASM) is a model for the formal, operational description of algorithms. The states of an Abstract State Machine are general mathematical structures. The inventor of the model is Yuri Gurevich. Egon Börger has further developed the ASM for practical application [34].

Abstract State Machines (ASM) are finite sets of transition rules of the form

If condition then action

with which the states of an ASM are changed. Condition is any logical expression and action any action. As a rule, action is a value assignment of the form f(t1,..... tn) := s. The meaning of the rule is to execute the specified rule in the current state if the specified condition is met in that state. ASM states are generally defined as arbitrary sets of arbitrary elements with arbitrary functions (operations) and predicates defined on them. In the case of business objects, the elements are placeholders for values of any type and operations such as creating, duplicating, deleting, or algebraically manipulating objects.

A calculation step of an ASM in a certain state means that all actions for which the condition is true are executed simultaneously. Simultaneous execution can abstract from irrelevant sequences.

Several ASMs can run simultaneously and be linked via so-called controlled or monitored functions. A given ASM M can update controlled functions, but it cannot be modified by other ASMs in its environment. Monitored functions of a given ASM M can only be updated by its environment. By using controlled and monitored functions in pairs, a network of parallel coordinating ASMs can be set up.

Object-Oriented Models

The computer science models considered so far focus either on data or functions. The representation of data in an ERM and functions in a flowchart complement each other only in decoupled representations. The distinction between data and function views in ARIS makes this clear. Object-oriented modeling no longer reduces these individual entities into their separate parts but considers them as an integrated whole in which the individual components are interconnected and interdependent.

An object-oriented model is a view of a complex system in which the system is described by the interaction of objects. This type of modeling is intended to reduce the complexity of the description of situations to be mapped in software. The object orientation considers the entities occurring in the real world as objects. A telephone is just as much an object as a bicycle, a person or an employee. Such objects in turn consist of other objects such as screws, rods, arms, feet, head, etc. As is usual in modeling, the objects are reduced to their properties that are significant in the respective situation. For example, an employee in a payroll accounting system is reduced to name, address, employee number, agreed income, tax class, and so on.

The objects considered in a model are not designed individually. A rough blueprint with similar properties is created for similar objects.

For example, one models the properties of books for a library application that are identical for all books. Such general descriptions of objects are called classes in object orientation. These classes are then used to create the required concrete objects (instances) within the model. The representation of reality in a model is therefore a two-stage process. First, similar objects of reality are identified and described as classes. The individual objects are then created as instances of a class. A class thus describes the structure of a set of similar objects. Figure 2.26 shows a class-object relationship according to which the book "Subject-oriented Process Management" is an instance of the class Book. The notation used comes from the Unified Modeling Language (UML). UML is a language standardized by the Object Management Group for the description of object-oriented models.

The properties of a class are

- their components and the data and information they contain, also called attributes,
- the operations defined on the components and their parameters (methods) with which an object can be manipulated or its status queried, and
- conditions, prerequisites, and rules that the objects must fulfill (constraints).

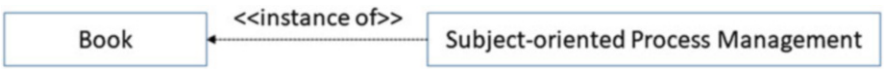

Figure 2.26: Relationship class-object

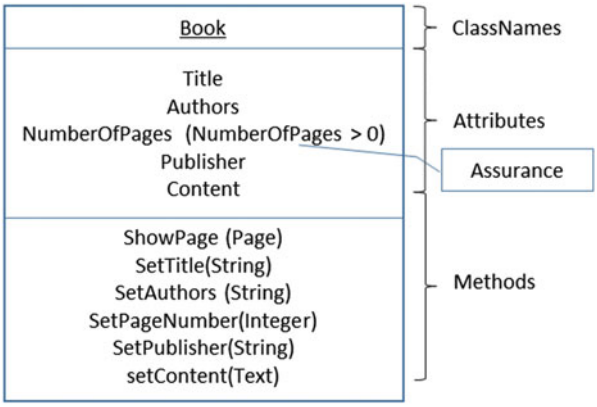

Figure 2.27: Description of the Book class

Figure 2.27 shows an example of the description of the Book class. This class has five attributes. The Page Number attribute has a constraint that the page number must be positive. The class allows the access and manipulation of its data with six methods. Thus, the title, the authors, the page number, the publisher and the content can be set (Set). The operation (method) DisplayPage(Page) can be used to "read" the contents of the specified page from the book.

Objects can now be instantiated from such a class definition. The operations can then be used to set the corresponding attributes for each of these instances.

An object-oriented model describes not only the definitions of the classes and the associated objects, but also the relationships between the classes. The types of relationships are:

- Inheritance

  Properties can be passed from one class to the next. This is called inheritance. The class that is used as the basis for inheritance is called the superclass, and the class that inherits is called the subclass. Thus, a class Notebook is a subclass of the class Book. The method "PageEntry(page, content)" is added to the Notebook class. This allows a text to be entered on the specified page; otherwise, the Notebook subclass inherits all attributes and methods from the Book class. Figure 2.28 shows the inheritance relationship between Book and Notebook. The triangular arrow points from the subclass to the superclass.

- Associations

  An association is a relationship between different objects of one or more classes. Associations are represented as a simple line between two classes. The line can be provided with a name (identifier) and number specification. The

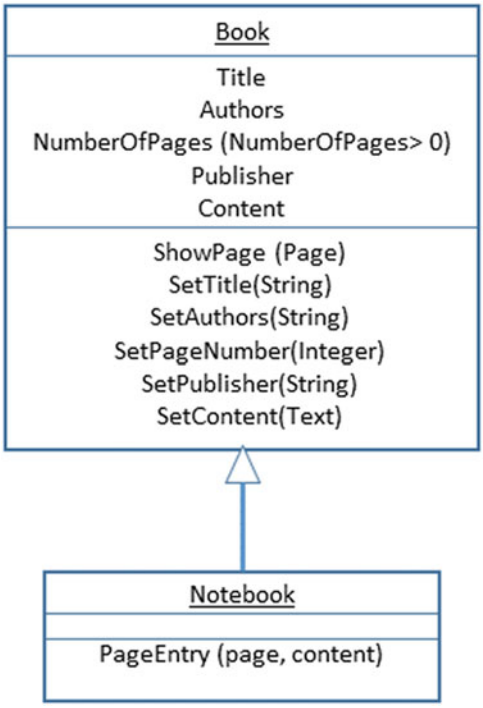

Figure 2.28: Inheritance of properties

associated classes can also receive names regarding the relationship. Figure 2.29 illustrates an example of an association. A person (employee) can be employed by no company or by just one company (employer) and a company can employ one or more persons. Associations are very similar to entity-relationship diagrams.

- Aggregations
  Aggregations are a variant of associations. This is also a relationship between two classes, but with the peculiarity that the two classes are related to each other like one part of a whole. An aggregation is made up of a quantity of individual parts. Figure 2.30 shows an aggregation example in which a company consists of departments and a department of employees.

- Composition
  A special form of aggregation is composition. Here the whole depends on the existence of its individual parts. Figure 2.31 shows a change in the above aggregation. A department does not exist anymore if no one belongs to it.

- Objects
  Communicate with each other, i.e., one object sends messages to another object. The messages then trigger the associated operations. An object therefore only understands the messages for which it contains the corresponding operations. Figure 2.32 shows a person's communication with the notebook to enter a new note.

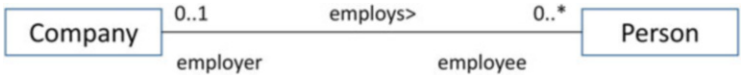

Figure 2.29: Associations between objects

Figure 2.30: Multilevel aggregation

Figure 2.31: Composition

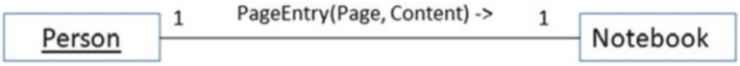

Figure 2.32: Message between objects

The described constructs form the nucleus of the object-oriented modeling approach. In addition, there are numerous enhancements for depicting certain circumstances, such as the so-called container classes.

2.7 Agent/Actor-Oriented Models

So far, there is no standardized and generally accepted definition of the agent concept. The term is defined in detail somewhat differently depending on the application domain [34, 35]. What all these definitions have in common, however, is that an agent is a scoped unit that is able to pursue the tasks assigned to it flexibly, interactively and autonomously. The term actor is often used as a synonym for the term agent. Thus [36] define a business actor as "... an entity that is capable of performing behavior". Multi-agent systems consist of several agents that exchange messages synchronously or asynchronously. Multi-agent systems can map the structures of software systems or serve as models for social systems. Depending on the application situation, a distinction can then be made between software agents and human agents.

In a way, the processes in CCS, the π Calculus and CSP can be seen as multi-agent systems. The term process is used there analogously to the terms agent and actor.

An agent-oriented model therefore contains the agents, the communication paths and the messages that are exchanged. A collection of agent-oriented modeling languages and the corresponding procedures can be found in [37, 38].

2.8 Conclusion: Models for Business Processes

Business process models depict the part of reality of business processes under consideration. The subjective understanding of the business process concept influences which process aspects are considered essential and therefore placed in the foreground in the models. The intentions and interests of the person creating the model are reflected here. Consequences are numerous interpretations of the business process concept, each of which is neither right nor wrong, but merely sets different focal points.

The following definitions of the term business process are examples of this:

1. "Sequence of value creation activities (value creation) with one or more inputs and a customer benefit generating output"[39].
2. "A process is the closed, temporal and logical sequence of activities that are necessary for the processing of a business-relevant object."[40]

Both definitions focus on the necessary activities and their consequences. The first example additionally mentions the input and the output with the customer benefit, while in the second definition the processing of the business objects is included instead.

On the other hand, neither definition includes the actors and necessary resources. They do not consider by whom and with what the activities are carried out. There is no relation to the organization in which a business process is embedded, or to which IT applications or other resources are required to execute it.

We therefore follow an understanding of the term based on that of Gerhard Schewe [41] that also takes these missing aspects into account [41]:

1. A process is the sum of linked activities (tasks)
2. carried out by actors (people, systems as task bearers)
3. in logical and chronological order
4. with aids (equipment, information)
5. for processing a business object
6. to satisfy a customer need (and thus contribute to value creation)
7. which includes a defined beginning and input
8. as well as a defined end and a result.

As already explained in Chapter 1, we will reorganize the components somewhat and group them as follows:

1. Process strategy: A process has

    a. a defined start and input (start event)
    b. and a defined end with a result
    c. that contributes to the satisfaction of a customer's needs (and thus to the creation of value)

2. Process logic: A process

   a. is the sum of linked activities (tasks)
   b. which, after the start event, are used by actors
   c. in logical and chronological order
   d. for processing a business object in order to
   e. generate the desired result.

3. Process realization: A process is realized

   a. with people and/or machines, that take over the tasks of the respective actor
   b. and carry these tasks out with tools (equipment, information, application programs, etc.).

With this understanding of business processes, the relationship between the various models from different domains described in this chapter and Business Process Management becomes clear. Figure 2.33 shows the associated integrative character of business process models.

Habermas's and Luhmann's models deal with aspects of social systems and organizations. They describe which components and relationships make up an organization and how people are positioned in it. Complex organizations can, for example, be structured into sub-organizations on the basis of operational functions, range of services, geographical aspects or combinations of these. The result is the organization chart.

Among other things, business models consider the aspects of customers, suppliers, partners and added value and thus look at the external service relationships on the one hand. On the other hand, they also establish the connection to the more

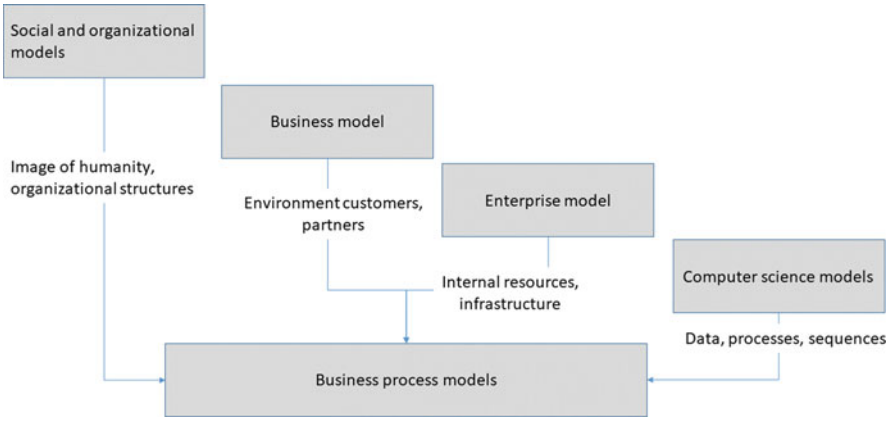

Figure 2.33: Integration of different models through business process models

inwardly oriented enterprise architecture, especially with value promises (products and services), activities and resources. At the business level, the organizational structure is modeled within this enterprise architecture with the personnel resources, the processes, and the logical business objects. The linkage with the technical layer of the enterprise architecture leads to the models from computer science. These describe, for example, data structures, control flows and algorithms for programs as well as the design and interaction of other information and communication technology components necessary for the execution of desired actions within the framework of process support and automation.

Models of business informatics generally try to unite computer science with models of social systems. These converge in business process models.

References

1. Gabriel, M. (2015). Why the world doesn't exist, Ullstein Taschenbuch.
2. Alle, V. (2015). *Value networks and true nature of collaboration*. Tampa, FL: Meghan Kiffer Press.
3. Stachowiak, H. (1973). *General model theory*. New York: Springer.
4. Kieser, A., & Walgenbach, P. (2007). *Organization*. Stuttgart: Schäffer-Pöschl.
5. Crowther, D., & Green, M. (2004). *Organizational theory*. London: Chartered Institute of Personnel Development.
6. Schreyögg, G. (2008). *Organization* (5th ed.). Wiesbaden: Gabler.
7. Kieser, A., & Ebers, M. (Eds.). (2006). *Organizational theories*. Stuttgart: Kohlhammer.
8. Szyperski, N. (1980). Information needs. In E. Grochla (Ed.), *Handwörterbuch der Organization* (2nd ed., p. 904). Stuttgart: C.E. Poeschel.
9. Krcmer, H. (2017). Information, encyclopedia of business informatics. *Online Lexicon.* Last accessed July 31, 2017, from http://www.enzyklopaedie-der-wirtschaftsinformatik.de/lexikon/daten-wissen/Informationsmanagement/Information-/index.html
10. Berghaus, M. (2004). *Luhmann made easy* (2nd ed.). Cologne: Böhlau.
11. Luhmann, N. (1984). *Social systems*. Frankfurt: Suhrkamp.
12. Scheer, C., Deelmann, T., & Loos, P. (2003). *Business models and internet-based business models – Definition and participant model*. Working Papers of the Research Group Information Systems & Management, University of Mainz
13. Wirtz, B. W., Schilke, O., & Ullrich, S. (2010). Strategic development of business models implications of the web 2.0 for creating value on the internet. *Long Range Planning, 43*, 272e290. Elsevier 2010.
14. Osterwalder, A., & Pigneur, Y. (2010). *Business model generation*. Hoboken: Wiley.
15. Siemoneit, O. (2010). A theory of science in business administration, Dissertation. University of Stuttgart.
16. Kieser, A., & Ebers, M. (2006). *Organizational theories*. Stuttgart: Kohlhammer.
17. Wieß, G., & Jakob, R. (2005). *Agent-oriented software development*. Berlin: Springer.
18. Brushwood, W. (2010, July 15). *Petri nets: Modeling techniques, analysis methods, case studies* (2010th ed.). Vieweg+Teubner Publishers.
19. ITIL. (2018). *Best practice*. Last Access February 2018, from https://www.axelos.com/best-practice-solutions/itil
20. Matthes, D. (2011). *Enterprises architecture compendium*. Heidelberg: Springer.
21. Lehner, F. (1995) Models and modeling in business informatics - Attempting to determine the current situation. In H. Wächter (Ed.), *Selbstverständnis betriebswirtschaftlicher Forschung und Lehre*. Wiesbaden: Gabler Publishing House.

22. Kaplan, R. S., & Norten, D. P. (1997). *Balanced scorecard*. Stuttgart: Schaeffer-Poeschl Publishers.
23. Robin Milner, A. (1980). *Calculus of communicating systems*, Lecture Notes in Computer Science. Springer.
24. Milner, R. (1999). *Communicating and mobile systems: The π-calculus*. New York: Cambridge University Press.
25. Hoare, C. A. R. (1978). Communicating sequential processes. *Communications of the ACM, 21* (8), 666–677.
26. Hoare, C. A. R. (1985). *Communicating sequential processes*. Englewood Cliffs, NJ: Prentice Hall.
27. Open Group. (2019). ArchiMate® 3.1 specification. https://pubs.opengroup.org/architecture/archimate3-doc/
28. Börger, E., & Stärk, R. (2003). *Abstract state machines*. Springer.
29. Lankhorst, M., et al. (2017). *Enterprise architecture at work* (4th ed.). Berlin: Springer.
30. Wooldridge, M. (2002). *An introduction to multi agent systems*. Chichester: Wiley.
31. Cossetino, M., et al. (2014). *Handbook on agent-oriented design processes*. Berlin: Springer.
32. Schewe, G. (2017). *Business process, Gabler business encyclopedia*. Last accessed July 31, 2017 from http://wirtschaftslexikon.gabler.de/Definition/geschaeftsprozess.html#definition
33. Becker, J. (2012, October). Business process modeling, encyclopedia of business informatics. *Online encyclopedia.* Last accessed July 31, 2017, from http://www.enzyklopaedie-der-wirtschaftsinformatik.de/lexikon/daten-wissen/Informationsmanagement/Information-/index.html
34. Schewe, G. (2017). *Business process, Gabler business encyclopedia.* Last accessed July 31, 2017 from http://wirtschaftslexikon.gabler.de/Definition/geschaeftsprozess.html#definition
35. Stähler, P. (2002, September). Business models as an unit of analysis for Strategizing. In P. Steels (Ed.), *Business models in the digital economy: Characteristics, strategies and effects*. Cologne: Josef Eul.
36. Habermas, J. (1981). *Theory of communicative action* (Vol. 1, 2014th ed.). Frankfurt am Main: Suhrkamp Paperback Science.
37. Römpp, G. (2015). *Habermas made easy*. Cologne: Böhlau.
38. Sowa, J. F., & Zachman, J. A. (1992). Extending and formalizing the framework for information systems architecture. *IBM Systems Journal, 31*(3), 590–616.
39. TOGAF. (2010, October). *TOGAF version 9*. Van Haren Publishing, Zaltbommel.
40. Franklin, S., & Graesser, A. (2017). *Is it an agent, or just a program? A taxonomy for autonomous agents.* Institute For Intelligent Systems, University of Memphis, Memphis, TN. Last accessed September 2017, from https://pdfs.semanticscholar.org/288d/7952b6648749fcbdcedabedf8f43cf7fda52.pdf
41. COBIT 5. (2018). Last Access February 2018, from http://www.isaca.org/COBIT/Pages/COBIT-5-german.aspx

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (http://creativecommons.org/licenses/by/4.0/), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.